HowTo Symbolic and Hard links

بسم الله الرحمن الرحيم

Seven_Eleven
لينكساوي

موضوعنا اليوم سيكون عن موضوع شيق وهو معتمد بشكل كبير على القدرة التى تمتلكها للتخيل !! سوف اتحدث اليوم عن ال hard links وال soft links على جنو/لينوكس . بالطبع إذا كنت مستخدما للويندوز من قبل فمن المحتمل أنك استخدمت الإختصارات أو فيما يعرف بال shortcuts ، هذا بالضبط ما أريده , فجنو/لينوكس يحتوى على إختصارات شبيهة بتلك التى يمتلكها ويندوز والتى تُعرف بإسم hardlinks و ال softlinks .

والآن لنبدأ مع الموضوع مباشرة وأول ما نتحدث عنه تعريف ال symbolic links or symlinks or softlinks ثلاثة مسميات لمعنى واحد :

تعتبر ال softlinks او ال symlinks أشبه ما يكون باختصارات الويندوز أو windows shortcuts . لأنها عبارة عن وصلة تحتوى على المسار الخاص بملف معين فى مكان ما دون الحاجة إلى تكرار ذلك الملف .

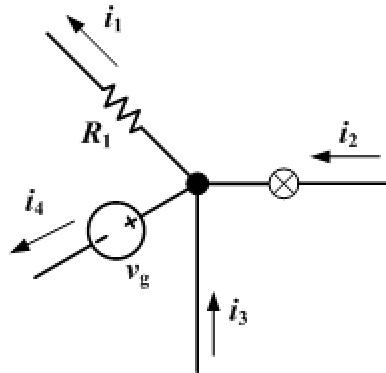
أما ال hard links فهى عبارة عن وصلة لملف ما على جنو/لينوكس تحتفظ بكل المعلومات الخاصة بذلك الملف وتكون أشبه ما يكون بعملية النسخ او copying ولكن مع وجود أختلافات جوهرية .

البعض سوف يصرخ عاليا ......!! لم أفهم ماذا تقصد بال hard links ؟؟!! حسنا تابع معى ولا تتعجل الأمور ، بعد قليل ستتضح الأمور من خلال الأمثلة العملية التى سوف نقوم بتطبيقها معا لاحقا .

والآن نستمر فى تعريف بعض المفاهيم والتى ذات صلة وثيقة بال hard links وال symbolic links ونبدأها معا :

أوٍلا : inode تنطق هكذا (eye-node) وهي اختصار لكلمتىّ information node والتى تعنى كترجمة حرفية عُقدة المعلومات !! لست أعنى بكلمة عُقدة أنها معلومات مُعقّدة أو مقياس سرعة السفن بالعٌقدة !!

كلمة عٌقدة تعنى فى عالم الإلكترونيات نقطة تتجمع حولها التيارات الكهربية الداخلة والخارجة منها وإليها ، ولعل بعضكم سمع بقانون كيرشوف للتيار الكهربى وهو مجموع التيارات الداخلة إلى node تساوى مجموع التيارات الخارجة منها !! استمتع بالصورة التالية :



الآن أسمع من يهمز بى ولسان حاله يقول هل الكاتب قد جُن ؟؟! ما علاقة ذلك بجنو/لينوكس ؟ من المثال السابق حاولت تقريب معنى node لك لكى تستطيع تخيل ما يفعله نظام الملفات أو ال filesystem على جنو/لينوكس لتخزين المعلومات التى تخص ملف أو مجلد معين .

نرجع الآن إلى تعريف ال inode بالنسبة لنظام الملفات على جنو/لينوكس ، حيث يتم تمثيل كل ملف على النظام ب inode ، تلك ال inode تحتوى على كل المعلومات التى تخص ذلك الملف بداية من الملف نفسه و مكان المعلومات المخزنة التى تخص ذلك الملف على الهارددیسك والتصاريح الخاصة بذلك الملف مرورا بنوع الملف أو the type of file إلخ ....

وصلنا الآن إلى صلب الموضوع ونبدأ أولا بالعلاقة بين كل inode وال hard links :

علاقة ال inode بال hard links علاقة وطيدة ، بمعنى كل inode على نظام الملفات أو ال filesystem تنقسم إلى hard link واحد أو اكثر من hard links . حيث يحتوى ال hard link الخاص بملف على كلاً رقم عُقدة المعلومات أو inode number ويحتوى أيضا على إسم الملف . وكما ذكرنا سابقا أن ال inode نفسها تحتوى على الملف ، وعلى التصاريح الموجودة عليه ، وكذلك نوع الملف .

ولذلك يستطيع نظام الملفات أو ال filesystem تحديد مكان ال inode بواسطة ال inode number والذى يوجد داخل ال hard link .

إلى الآن أرى العيون جاحظة لم ترى فائدة لل hard links ؟؟ اليس كذلك ؟

تريث قليلا وتابع القراءة ، من الممكن للملف الواحد أن يمتلك أكثر من hard link ، وذلك يعنى أنه من الممكن أن توجد عدة ملفات بأسماء مختلفة تشير إلى نفس الملف ، والتى تتملك بدورها هذه الملفات ذات الأسماء المختلفة نفس ال inode number للملف الأصلى الذى تم إنشاء منه عدة hard links .

ولذلك وانتبه معى ، ان الفائدة الجوهرية لل hard links تكمن فى امتلاك الملفات ذات الأسماء المختلفة لنفس ال inode number للملف الأصلى ، وبمعنى أوضح نظرا لأن ال inode number للملف الأصلى هو نفسه ال inode number لل hard links فإنه تصبح لدينا القدرة على تعديل مثلا الملف الأصلى وينتج عن ذلك نفس التعديل فى الملفات الأخرى دون الحاجة إلى تعديلهم منفصلين ، ونتيجة ذلك توفير الكثير من الوقت والجهد إذا كانت عملية التعديل تحتاج إلى وقت كبير .

ملحوظة : لا يمكنك عمل hard links على عدة أنظمة ملفات مختلفة ، فلابد لكل ال hard links التى تخص

ملف معين أن تكون على نفس نظام الملفات أو the same filesystem ، وذلك لأن لكل نظام ملفات أو filesystem مجموعة من ال inodes التى تخصه ، ولذلك من المحتمل وجود أرقام لعقُد المعلومات أو inode numbers مكررة مع نظام ملفات آخر .

الآن إلى التطبيق العملى للجزء الخاص بال hard links قم معى بفتح الطرفية ونفذ التالى :

كود:

```
debian:~# cd; echo "hello" > firstlink
```

ولنشرح ما نفذنا فى الطرفية :

الأمر cd يقوم بتغيير الدليل إلى المجلد الذى يتم كتابته بعده وهنا قمنا بكتابة الأمر cd فقط والذى يشير عند كتابته وحده إلى تغيير الدليل الحالى إلى المجلد ال home ثم بعد ذلك استخدمنا ال ; أو Semi-Column والتى تستخدم للفصل بين أكثر من أمر عند تنفيذ عدة أوامر على التوالى فى الطرفية ، ثم بعد ذلك استخدمنا الأمر echo والذى يستخدم لطباعة نص على ال stdout أو فيما يسمى بال standard output والتى تكون فى حالتنا هذه هى الطرفية ، ثم بعد ذلك استخدمنا العلامة > لعمل redirect لخرج ال echo ليتم وضعه داخل ملف يتم إنشائه تحت إسم first link .

الآن لنتأكد فعلا من إنشاء الملف firstlink ووجود الكلمة hello بداخله عن طريق الأمر التالى :

كود:

```
debian:~# cat firstlink

hello
```

بعدما تأكدنا فعلا من إنشاء الملف firstlink والتأكد من محتوى الملف ووجود كلمة hello سنقوم الآن بإنشاء hard link من الملف باستخدام الأمر ln كالتالى :

كود:

```
debian:~# ln firstlink secondlink
```

ولنتأكد الآن أيضا من إنشاء الملف secondlink والتأكد من إحتوائه على نفس الكلمة التى توجد فى الملف firstlink وهى كلمة hello عن طريق الأمر cat كما ذكرنا سابقا :

كود:

```
debian:~# cat secondlink

hello
```

الآن تم إنشاء hard link من الملف firstlink تحت إسم secondlink وكما ذكرنا سابقا سيمتلك الملف secondlink نفس ال inode number الخاصة بالملف firstlink وللتأكد من ذلك قم بتنفيذ الأمر التالى :

كود:

```
debian:~# ls -il
```

كود:

```
315 -rw-r--r-- 2 root root 13 2007-12-02 10:21 firstlink
315 -rw-r--r-- 2 root root 13 2007-12-02 10:21 secondlink
```

كما تلاحظ الرقم الأول من اليسار وهو 315 هو الذى يمثل ال inode number وفى حالتنا هذه نجد أنه فعلا لدى الملفان firstlink و secondlink نفس ال inode number ، ونلاحظ أيضا وجود نفس التصريحات الخاصة بكل ملف واحدة ، أما الرقم 2 فيدل على عدد ال hard links الخاصة بالملفان .

طيب ننتقل الآن إلى الفائدة الجوهرية التى ذكرناها بأعلى وهى عندنا نعدل ونغير فى محتويات ملف هل سيتم فعلا تغيير وتعديل محتويات الملفات الأخرى أم لا ؟؟ لنرى سويا ولنقم بتنفيذ التالى فى الطرفية :

كود:

```
debian:~# echo "chagne" >> secondlink
```

هنا قمنا مرة أخرى باستخدام الامر echo لطباعة الكلمة change ووضعها داخل الملف secondlink ولكن مع الإحتفاظ بمحتويات الملف والعلامة >> هى التى قامت بهذه المهمة أى التعديل على الملف ووضع الكلمة change دون حذف الكلمة hello .

ملحوظة : لكى تلمس الفارق ما بين العلامة > والعلامة >> قم بتنفيذ الأمر السابق مرة أخرى ولكن باستخدام العلامة فقط ليكون شكله كالتالى :

كود:

```
debian:~# echo "change" > secondlink
```

نرجع مرة أخرى لنلاحظ هل حدث تغيير فعلا فى محتويات الملف secondlink وبالتالى هل حدث نفس التغيير فى محتويات الملف firstlink أم لا ونستخدم أيضا الامر cat فى هذه المهمة كالتالى :

كود:

```
debian:~# cat secondlink

hello
change
```

ثم مرة اخرى الأمر cat مع الملف firstlink ليكون كالتالى :

كود:

```
debian:~# cat firstlink

hello
change
```

نجحنا !! فعلا كلاً من الملفين احتوى على نفس الكلمتين كلمة hello وكلمة change وحدث ذلك بالتغيير فى الملف secondlink دون الحاجة إلى التغيير فى الملف firstlink هنيئاً ! .

وإلى تساؤل آخر هل لو غيرنا التصاريح الموجود على hard link سيتم تغيير نفس التصاريح على ال hard links الأخرى أم لا ؟ لنتابع معا ونقوم بتنفيذ التالى :

بالبداية نقوم باستعراض التصاريح الموجودة حاليا على الملف باستخدام الأمر ls مع الخيار l- كالتالى :

كود:

```
debian:~# ls -l secondlink

-rw-r--r-- 2 root root 7 2007-12-02 11:39 secondlink
```

ثم نقوم بتغيير التصاريح الموجودة على الملف باستخدام الأمر chmod كالتالى :

كود:

```
debian:~# chmod 777 secondlink
```

ملحوظة : ليس بالضرورة تطبيق التصريح 777 على الملف ، أنا ما استخدمت ذلك التصريح إلا للتوضيح فحسب فيمكنك وضع أى تصريح يعجبك !

ثم نستعرض التصاريح على الملف مرة اخرى باستخدام الأمر ls مع الخيار l- كالتالى :

كود:

```
debian:~# ls -l secondlink

-rwxrwxrwx 2 root root 7 2007-12-02 11:39 secondlink
```

نجد أن التصاريح قد تغيرت على الملف ، والآن نتأكد من التصاريح على الملف firstlink هل هى نفس التصاريح على الملف secondlink أم لا بتنفيذ التالى فى الطرفية :

كود:

```
debian:~# ls -l firstlink

-rwxrwxrwx 2 root root 7 2007-12-02 11:39 firstlink
```

بالفعل تغيرت التصاريح !! وأصبحت واحدة دون المساس بالملف firstlink وهذا يعنى أن التصاريح فعلا يتم تخذينها فى ال inode وليس فى ال links !!

والآن إلى نقطة أخرى وهى هل إذا تم حذف ملف من أو حذف hard link من ال hard links سيتم حذفهم جميعا أم حذف ال hard link المطلوب فقط ؟؟

للإجابة على ذلك التساؤل نقوم بتنفيذ الأمر التالى فى الطرفية :

كود:

```
debian:~# rm firstlink
```

ثم نقوم باستعراض محتويات المجلد مرة أخرى باستخدام الأمر ls مع الخيار l- كالتالى :

كود:

```
debian:~# ls -l

-rwxrwxrwx 1 root root 7 2007-12-02 11:39 secondlink
```

لاحظ تم حذف الملف firstlink دون حذف الملف secondlink أى أن عملية حذف hard link غير معتمدة على ال hard links الآخرى كما تلاحظ أيضا أن عدد ال hard links تم تخيفضه من الرقم 2 إلى الرقم 1.

والآن إذا قمنا بحذف الملف الآخر وهو secondlink عن طريق الأمر rm كالتالى :

كود:

```
debian:~# rm secondlink
```

الآن أصبح لا يوجد hard links لأين من الملفين firstlink والملف secondlink حينها يقوم جنو/لينوكس بحذف الملف نفسه أى حذف ال inode .

symlinks

بعد أن انتهينا من الجزء الأول والذى قمنا بتفصيل بعض الأمور عن ال hard links ننتقل الآن إلى الجزء الثانى من الوصلات أو ال links ألا وهى الوصلات المرنة أو soft links or symbolic links or softlinks كما قلنا فى أول الموضوع ثلاث مسميات لمعنى واحد .

وكما ذكرنا سابقا أن ال symlink عبارة عن ملف خاص يحتوى على العنوان أو المسار الخاص بملف معين على نظام الملفات أو filesystem ونستنبط من ذلك أنه يمكن عمل symbolic link من أى hard link لأنه ال hard link يكون إما الملف الأصلى نفسه أو link من ملف له نفس الخصائص للملف الأصلى أو بمعنى مٌبسط ال soft link ماهو إلا عبارة عن اختصار أو shortcut من الملف ليس إلا ، ولذلك عندما تتصفح محتويات symbolic link فأنت تتصفح محتويات الملف الأصلى الذى يشير إليه ال symlink وليس محتويات ال symlink نفسه ، ولذلك باستخدام ال soft link يمكنك عمل وصلات أو links لكل من المجلدات أو directories ، كذلك الأجهزة أو ال Devices مرورا بال symlinks نفسها لأن كل ذلك فى النهاية ماهو إلا عبارة عن ملفات بالنسبة لنظام التشغيل جنو/لينوكس .

وبعد أن استعرضنا تعريف مُبسط لل symlinks ، الآن سنقوم بذكر بعض الأمور التى يختلف فيها كلاً من ال hard links والsymlinks :

* ذكرنا أن ال hard link يحتوى على كل من إسم الملف أو filename بالإضافة إلى ال inode number الخاص بذلك الملف وذكرنا أن ال inode هى نفسها الملف والتى تعنى مكان تخزين الملف على القرص الصلب والتصاريح الموجودة وكذلك نوع ذلك الملف ، بينما ال symlink عبارة عن inode تحتوى على إسم ال hard link .

* كل ال hard links الخاصة بنفس الملف لها نفس الحالة ، ولذلك عند إجراء أى عملية عن أين من تلك ال hard links ، فتبعا لذلك يحدث نفس التغيير بسبب تلك العملية على باقى ال hard links الخاصة بتلك الملف وذلك يحدث لأن كل ال hard links لنفس الملف تمتلك نفس ال inode ، أما عند إجراء أى عملية على أى symlink فالتغيير يحدث إما فى ال inode الخاصة بال symlinks نفسه والذى يكون تغيير فى إسم ال hard link أو يحدث التغيير فى ال hard link الذى يشير إليه ال symlink .

* ال symlinks نستطيع إنشائها على أكثر من filesystem لأنها كما ذكرنا تحتوى على الإسم الكامل أو المسار الكامل الخاص بال hard link ابتداءً من ال root directory ، بينما لا نستطيع إنشاء أكثر من hard link على أكثر من نظام ملفات أو filesystem وذلك لأن ال hard links لنفس الملف تمتلك نفس ال inode number وال inode numbers لنظام الملفات الواحد تكون وحيدة أو unique ولذلك عن إنشاء hard links على أكثر من نظام ملفات قد تتشابه ال inode number الخاصة ب filesystem ب inode number خاصة ب filesystem آخر .

* تستطيع عمل symlinks للمجلدات أو ال directories ولكن لا تستطيع عمل hard links لهذه المجلدات وذلك لأن كل مجلد بذاته يحتوى على hard links مختلفة وكل مجلد فرعى تحت ذلك المجلد الرئيسى أو ال parent directory يحتوى على hard links مختلفة وخاصة به وهكذا ... وبالتالى يمكن عمل symlink لكل من الملفات و المجلدات بينما يمكن عمل hard links للملفات فقط .

* حذف ال symlink لا يؤثر على الملف الأصلى فى شىء وبالتالى ما يُحذف فى حالة ال symlink هى الوصلة فقط أو ال link ، أما حذف hard link الوحيد الخاص بملف ما كفيل بحذف الملف نفسه من على نظام الملفات .

بعد أن استعرضنا بعض الفروقات بين كل من ال symlinks وال hard links نتطرق إلى الجزء التطبيقى الخاص بال symlinks ونبدأ اولا بفتح الطرفية وتطبيق الأمر التالى :

كود:

```
cd; ln -s /tmp/me MyTmp
```

شرح الأمر :

فى البداية قمنا بتغيير الدليل أو المجلد الحالى الذى نعمل عليه إلى مجلد ال home directory بإستخدام الأمر cd فقط ، بعد ذلك العلامة Semi-column أو ال ; والتى تستخدم لتنفيذ عدة أوامر على التوالى فى الطرفية ، ثم بعد ذلك استخدمنا الأمر ln مع الخيار s- لعمل وصلة أو link ولكن هذه المرة من نوع symlink أو softlink وذلك للمجلد الفرعى me الموجود على مجلد الملفات المؤقتة tmp/ تحت إسم للوصلة MyTmp .

الآن لنتأكد من إنشاء ال symlink للمجلد tmp/me/ تحت الإسم MyTmp وذلك باستخدام الأمر ls مع الخيار l- لعرض بعض المعلومات الأخرى عن الوصلة كالتالى :

كود:

```
debian:~# ls -l MyTmp
```

ليكون الخرج بالشكل التالى :

كود:

```
lrwxrwxrwx 1 muhammad muhammad 7 2007-12-04 09:13 MyTmp -> /tmp/me
```

لاحظ : فى بداية التصريح يوجد الحرف l والذى يدل فى حالة وجوده على أن الملف من نوع symbolic link ولمزيد من المعلومات عن أنواع الملفات :

كود:

```
d    directory

l    symbolic link

b    block device

c    character device

p    named pipe

s    socket
```

لاحظ أيضا : أننى قمت بعمل symbolic link لمجلد وهمى وهو فى حالتنا هذه me !! وللتأكد من ذلك قم بتنفيذ الأمر التالى فى الطرفية :

كود:

```
debian:~# ls -l /tmp/me

ls: /tmp/me: No such file or directory
```

ولذلك من ضمن الإشياء التى لابد أن تلاحظها أنك لو قمت بتغيير التصاريح الموجودة على الوصلة MyTmp فلن تتغير وستظل كما هى أى ستكون فى كل مرة rwxrwxrwx ولكن التغيير سيكون للملف الأصلى وللتأكد من ذلك استخدم الأمر chmod مع أى تصريح يعجبك :

كود:

```
debian:~# chmod 600 MyTmp
```

ثم نفذ الأمر ls مع الخيار l- مرة أخرى للتأكد من ذلك كما يلى :

كود:

```
debian:~# ls -l MyTmp

lrwxrwxrwx 1 root root 7 2007-12-04 08:56 MyTmp -> /tmp/me
```

كما تلاحظ لم تتغير التصاريح الموجودة على الوصلة MyTmp لأنه أساس الوصلة المجلد الفرعى me لا يوٌجد بالأساس !! والآن نقوم بإنشاء المجلد الفرعى me لكى نكرر ما فعلنا سابقا بالشكل التالى :

كود:

```
debian:~# mkdir /tmp/me
```

ثم نقوم بتنفيذ الأمر ls مع الخيار l- على المجلد لاستعراض التصاريح الخاصة به ليكون الناتج بالشكل التالى :

كود:

```
debian:~# ls -dl /tmp/me

drwxr-xr-x 2 root root 4096 2007-12-04 10:10 /tmp/me
```

كما تلاحظ ال symbolic link لها تصريح مختلف تماما عن المجلد الأصلى ، لنحاول الآن تغيير التصاريح على كل من الوصلة MyTmp ونرى ما التغيير الجوهرى الذى سوف يحدث :

كود:

```
debian:~# chmod 500 MyTmp
```

ثم بنقوم بعرض التصاريح باستخدام الأمر ls مع الخيار l- مرة أخرى :

كود:

```
debian:~# ls -l MyTmp

lrwxrwxrwx 1 root root 7 2007-12-04 08:56 MyTmp -> /tmp/me
```

لم تتغير التصاريح على الوصلة MyTmp وهذا يؤكد الكلام الذى ذكرناه سابقا أن التصاريح لا ولن تتغير ، ولنستعرض الآن التصاريح على المجلد me كما يلى :

كود:

```
debian:~# ls -ld /tmp/me

dr-x------ 2 root root 4096 2007-12-04 10:10 /tmp/me/
```

كما رأيت التغيير الذى حدث كان للمجلد الأصلى me وليس للوصلة MyTmp .

لكن قد يتساءل البعض ألا تعتبر التصاريح الموضوعة على ال symlinks وسيلة لتشكيل خطر على جنو/لينوكس وذلك لأنها باستمرار تحمل التصريح rwxrwxrwx ؟؟

الإجابة على ذلك التساؤل بسيطة لو ركزت قليلا فيما ذكرناه ان ال symlinks مجرد وصلة تحتوى على إسم الملف الأصلى فقط ولذلك التصاريح التى تمتكلها لا تمت بصلة للتصاريح الموضوعة على الملف الأصلى إلا إذا كانت التصاريح على الملف الأصلى وال symlink واحدة .

الآن إذا حاولت إنشاء ملف نصى داخل الوصلة MyTmp فمن البديهى أن الملف يتم إنشائه بشكل واقعى داخل المجلد tmp/me/ وليست الوصلة ذاتها ولنرى ذلك كيف يتم عمله كالتالى :

كود:

```
debian:~# touch MyTmp/myfile.txt
```

ثم نستخدم الأمر ls لعرض محتويات المجلد me فى الطرفية بالشكل التالى :

كود:

```
debian:~# ls /tmp/me

myfile.txt
```

بالفعل تم إنشاء الملف myfile.txt داخل المجلد me والآن إلى نقطة أخرى وهى لو قمنا بحذف الوصلة MyTmp هل سيتم حذف المجلد me أم لا ؟؟ لنرى سويا :

كود:

```
debian:~# rm MyTmp
```

لاحظ أننى قمت باستخدام الأمر rm وليس الأمر rmdir لأن ال symlink ما هى إلا ملف عادى وللتأكد من أن المجلد me مازال موجودا نقوم بعمل التالى :

كود:

```
debian:~# ls -al /tmp

dr-x------  2 root     root     4096 2007-12-04 10:31 me
```

فعلا المجلد me مازال موجودا ولم يتم حذفه ، الآن لنحذف الملف myfile.txt والمجلد me على الترتيب بالشكل التالى :

كود:

```
debian:~# rm /tm/me/myfile.txt; rmdir /tmp/me
```

الآن تم التخلص من كل الأشياء التى تخص المجلد me بداية من ال symlinks انتهاءً بمحتوياته .

أرجو من الله العلى القدير أن أكون وُفقت فى شرحى هذا فإن أصبت فمنه سبحانه وتعالى وإن أخطأت فمنى والشيطان وأتمنى إذا كان فيه أى إضافات أو تنقيحات تُثرى الموضوع فلا تبخلوا علينا بها .

السلام عليكم ورحمة الله وبركاته

St0rM-MaN
:: خبير برمجة ::

اقتباس:

بالمناسبة من هنا جت كلمة unlink فى بعض اللغات عشان حذف الملفات ^_^

غير كده وكده انا مش قريت الموضوع الصراحه
اي ملف في ال linux ماهو الا hard link للملف الاصلي والاخ seven_eleven وضح كده بس باسلوب تاني
كل ملف علي اللينكس الكيرنيل بيتعامل معاه عن طريق منشاة بيانات تسمي file موجوده في الكيرنيل ذات نفسه
بتحتوي علي العمليات المسموحه لذلك الملف .... الي اخره
ال hard link بيفتح structure جديده للملف دوت مع نفس ال inode
يعني عدة منشأت بتتعامل مع نفس ال inode
جميل عند الحذف ! الي بيحصل ان عدد ال منشأت بيقل علي حسب ال hard links او نقدر نقول ال open descriptors الي هي ارقام بتمثل الملفات المفتوحه حاليا علي جنو
لما بيوصل عددها لل 0 وفققفط لل 0 بيتم حذف الملف فعليا من علي وحدة التخزين
ولذلك لو في open descriptor الملف مش بيتمسح ابدا الا لما اخر descriptor يتقفل فعلا..
واحد حدق بقي يقولي" انت تاخل تتفذلك مثلا " ؟؟؟
حقوله لا انا الي عملته دوت عشان ححط حركه صغيره كده :
تخيل نفسك بتكتب برنام معين ومحتاج ملف مؤقت تكتب فيه الحاجات المهمه بتاعتك ؟
ممكن تعمل الملف دوت create وتقوم ماسحه عن طريق unlink بس تخلي ال file descriptor مفتوح :
d ساعتها انت بس الي حتقدر تقرأ من الملف دوت او تكتب فيه
في حين ان لو حد دخل في مكان التنفيذ وعمل ls -al حتي مش حيلاقي الملف دوت

Seven